Comparing and Fusing Terrain Network Information

Emmanuel Navarro, Bruno Gaume, Henri Prade

Comparing and fusing terrain network information

Emmanuel Navarro†, Bruno Gaume‡, and Henri Prade†

†: IRIT, Université de Toulouse III,
118 Route de Narbonne; 31062 Toulouse Cedex 9, France
E-mail: navarro@irit.fr, prade@irit.fr

‡: CLLE-ERSS, Université de Toulouse II,
5, allées Antonio Machado; 31058 Toulouse Cedex 9, France
E-mail: gaume@univ-tlse2.fr

Abstract. Terrain networks (or complex networks) is a type of relational information that is encountered in many fields. In order to properly answer questions pertaining to the comparison or to the merging of such networks, a method that takes into account the underlying structure of graphs is proposed. The effectiveness of the method is illustrated using real linguistic data networks and artificial networks, in particular.

1 Introduction

Complex networks [1, 19] are graphs with non-trivial topological features. In the following we prefer to call them "terrain networks" to emphasize the fact that they represent practical data, supposed to have some underlying structure. Moreover, it is a counterpart of the French "graphe de terrain". Such networks can be observed in many areas ranging from computer sciences to biology, linguistics, and social sciences. Examples of such graphs are synonymy networks between words, social relation networks between people, or protein interaction networks. One of their main features is to be globally sparse and locally dense. In other words, while their number of edges is relatively small, they exhibit a rather high transitivity (or clustering) coefficient (defined by the ratio of the number of 3-cliques over the number of paths of length 2). Moreover their diameter, i.e. the average minimal path length between pairs of vertices, is very small [19] and the degree distribution follows approximately a power law [1].

Since terrain networks are more and more common pieces of information, general information processing issues, such as comparison or fusion of two networks, make sense for them and become increasingly important. In this paper, we consider the particular case of special interest where the two graphs have the same vertices. This means that the two graphs represent data pertaining to the same items, objects, or agents. Generally speaking, the comparison of graphs may be envisaged in different ways. One may compare two graphs either at the edges and vertices level [8, 11, 16, 18, 20], or in terms of global structural property measures [10, 12]. None of these two classes of methods appear to be fully satisfactory for comparing terrain networks sharing the same vertices. Indeed, the former do not take into account the latent similarity information since they

work in a too local way, while the latter only deals with global properties without any reference to the fact that the graphs share common vertices.

Terrain networks depart from other graphs often encountered in AI. Indeed, graph representations are associated with taxonomies or ontologies, or with Bayesian nets. They encode various forms of generic knowledge, possibly pervaded with uncertainty, which can be applied to factual pieces information describing the particular situations to reason about. This contrasts with terrain networks which gather what may be called data information. They are made of collections of pieces of factual information, but we are no longer primarily interested in just answering requests pertaining to particular individuals. The emphasis is rather on the way the pieces of information are related together and are organized in cluster-like structures. Thus, for instance, the proximity between two graphs is not only a matter of identity of edges, but should also take into account the neighborhood structures of vertices. For example, a non-edge may "virtually" exist as an edge if there are short paths linking its vertices.

In this paper, we propose a general procedure that labels each pair of vertices in a graph, i.e., each edge, as well each non-edge, in terms of two categories: the edge (or the non-edge) is "confirmed", or is "not-confirmed" (in Section 2). Thus, the existence, or the non-existence of an edge between two vertices is confirmed, or not according to their neighborhood situation that in some sense support or not this existence, or non-existence. Then, we show the interest of such labeled graphs for comparing (in Section 3) or merging (in Section 4) terrain network information. Related work is discussed in Section 5.

2 What a data information graph may mean

In this section, data information graphs, issued from terrain networks, are considered as knowledge representation entities, which can be manipulated in order to lay bare some hidden part of the information. In such graphs, the information conveyed is not just made of a collection of links existing between certain pairs of vertices, but should also take into account the graph topology in the neighborhood of pairs of vertices. Before presenting a labeling procedure whose purpose is to confirm (or not) each edge and each non-edge in a graph in order to bring back the graph topology information, we first restate general knowledge representation concerns by examining in what respect a graph may be correct or complete.

2.1 Correctness and completeness of a graph

If the information given by a graph is correct and complete, any edge expresses the certainty of the existence of a relation between the two associated vertices, and the absence of edge between two vertices asserts that there is no relation between them. However, if a graph is only correct, each edge is there for sure, but the absence of an edge may be as much the result of missing information as acknowledging the certainty of the absence of link. Conversely, if a graph is only complete, no edge are missing, but some may be questionable. Then the absence of an edge reflects the certainty that there is no relation.

Also note that in case some prior knowledge exists about the graph, it may be used for revising it. Thus knowing, for instance, that the graph should represent a transitive relation, two situations would be of interest. If the graph is correct but incomplete, then it can be replaced by its transitive closure. If the graph is incorrect but complete, we may try to remove a minimal number of edges to make the relation transitive (but in general the solution is not unique!). However, in the following we do not assume the availability of such strong prior knowledge.

When comparing or merging two graphs, assuming that the information conveyed by each of them is correct, and/or complete is a crucial issue. Indeed in such operations, knowing of which edge, or non-edge one may be certain is clearly important. When a graph is correct and complete, any edge (resp. non-edge) is certain and has a status denoted $1!$ (resp. $0!$). When a graph is incorrect and incomplete, any edge (resp. non-edge) is uncertain and has a status denoted $1?$ (resp. $0?$). Table 1 sums up the four possible cases. More generally, the status of edges or non-edges in a graph may differ from one pair of vertices to another. Indeed it may be interesting to have such a binary "uncertainty" information for each edge and non-edge. Thus, for instance, a graph may be complete and correct, except for some pairs of vertices.

Table 1. Four possible cases of graph correctness and completeness, and there counterpart in terms of edges and non-edges certainty.

	edge	**non-edge**
correct *and* complete	$1!$	$0!$
incorrect *but* complete	$1?$	$0!$
correct *but* incomplete	$1!$	$0?$
incorrect *and* incomplete	$1?$	$0?$

In a similar spirit, in the next section, we propose a method for providing a similar type of status to each edge, or non-edge in a graph, and thus laying bare information that is not explicitly given with the graph. According to the neighborhood (possibly taken in a broad sense) of each pair of vertices, the corresponding edge (resp. non-edge) will be labeled $1?$ (resp. $0?$) and regarded as "uncertain", or will be labeled $1!$ (resp. $0!$) and regarded as "confirmed". Mind however that this is not genuine uncertainty information, but rather a way to bring back some "global information" to a local level. Indeed, roughly speaking the idea is to label with $0?$ the non-edge that are inside clusters, and to label with $1?$ the edges outside clusters, thus acknowledging the "imperfect transitivity" that may exist in the graph (and which is at work in the clusters).

2.2 Labeling edges and non-edges for reflecting the graph topology

In a graph, two vertices may be regarded as being "close" according to the graph topology between them, independently of the existence or not of a direct edge between them. For example, in the Figure 1 the pair a is not an edge, but the two vertices are close in the graph in the sense that there are 3 paths of length 2 between them. This contrasts with the situation of the non-edge b. Conversely the pair d is an edge, but the two vertices are relatively distant since there is no path between them other than this edge itself.

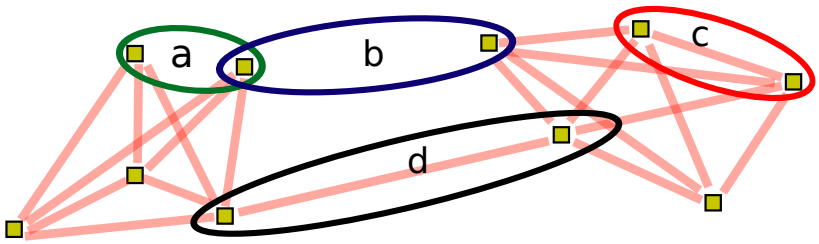


Fig. 1. A toy example of non-edge labeled "0?" (a) or "0!" (b), and edges labeled "1?" (d) or "1!" (c)

Lastly, the edge c is "strengthened" by the existence of 3 paths of length 2 between its two vertices.

The above observation is important when comparing and fusing two graphs (the problems considered in the next sections). Indeed, if a pair of vertices is an edge in one graph but not in the other one, the situation is not the same if this edge is like d or like c in Figure 1 (and similarly for the non-edge, if it is like a or like b). So, we propose to label each pair of vertices according to their closeness to be judged from the topology of the graph in the neighborhood of the two vertices, using the conventions summarized in Table 2. Several ways of evaluating closeness may be considered.

Table 2. Labeling procedure of edges and non-edges, according to a closeness evaluation of pairs of vertices

edge	closeness	**label**	
0	0	0!	not an edge, and not close in the graph.
0	1	0?	not an edge, *but* close in the graph.
1	0	1?	an edge, *but* not close in the graph.
1	1	1!	an edge, and close in the graph.

Evaluating closeness We now describe two methods that one may think of for evaluating closeness of a pair of vertices on an undirected graph $G = (V, E)$ (with V the vertex set and E the edge set).

Triangle A very simple method could be to consider as "close" every pair of vertices that are connected by a path of length 2. An edge will be confirmed (i.e. 1!) if it is supported by at least one path of length 2 ; or "unconfirmed" if there is no path of length 2 between the two corresponding vertices (i.e. 1?). Similarly a pair of non adjacent vertices, will be labeled as unconfirmed (i.e. 0?) if they are connected by at least one path of length 2, or as "confirmed" (i.e. 0!) if they are not connected by a path of length 2.

Table 3. Number and proportion of each label for edges and non-edges on 3 different terrain networks ($V.rob$, $V.pwn$, $V.wikt$) and one random network (rob^R, Erdös Rényi random graph of the same size than $V.rob$, average on 20 realizations)

		$V.rob$		$V.pwn$		$V.wikt$		rob^R	
triangle	1!	20442	76.9%	37473	91.6%	2886	34.8%	187.0	0.7%
	1?	6125	23.1%	3446	8.4%	5407	65.2%	26380.0	99.3%
	0?	395555	1.5%	527336	0.8%	33685	0.1%	190910.8	0.7%
	0!	26636924	98.5%	65884901	99.2%	26884813	99.9%	26841568.2	99.3%
5-confl.	1!	22726	85.5%	36760	89.8%	2864	34.5%	4032.2	15.2%
	1?	3841	14.5%	4159	10.2%	5429	65.5%	22534.8	84.8%
	0?	2844964	10.5%	3744489	5.6%	250177	0.9%	4795066.4	17.7%
	0!	24187515	89.5%	62667748	94.4%	26668321	99.1%	22237412.6	82.3%
10-confl.	1!	23143	87.1%	36887	90.1%	2980	35.9%	4657.2	17.5%
	1?	3424	12.9%	4032	9.9%	5313	64.1%	21909.8	82.5%
	0?	5282868	19.5%	7350176	11.1%	513214	1.9%	8292114.4	30.7%
	0!	21749611	80.5%	59062061	88.9%	26405284	98.1%	18740364.6	69.3%
20-confl.	1!	22405	84.3%	36741	89.8%	3056	36.9%	39.8	0.1%
	1?	4162	15.7%	4178	10.2%	5237	63.1%	26527.2	99.9%
	0?	8055282	29.8%	12241791	18.4%	948772	3.5%	10567375.4	39.1%
	0!	18977197	70.2%	54170446	81.6%	25969726	96.5%	16465103.6	60.9%

Confluence Short length random walks may provide a more accurate method for measuring the closeness of two vertices in a graph [6, 7, 13]. Let $G = (V, E)$ be an undirected and reflexive[1] graph. Let us imagine a walker wandering on G:

- At a time $t \in \mathbb{N}$, the walker is on one vertex $u \in V$;
- At time $t + 1$, the walker can reach any neighboring vertex of u, with a uniformly distributed probability.

This process is called a simple random walk [3]. It can be defined by a Markov chain on V with a $|V| \times |V|$ transition matrix $[G]$:

$$[G] = (g_{u,v})_{u,v \in V}, \quad \textbf{with} \quad g_{u,v} = \begin{cases} \dfrac{1}{d_G(u)} & \textbf{if } (u,v) \in E, \\ 0 & \textbf{else.} \end{cases}$$

where $d_G(u) = |\{v \in V / (u,v) \in E\}|$ is the degree of vertex u in the graph G. Since G is reflexive, each vertex has at least one neighbor (itself) thus $[G]$ is well-defined. Furthermore, by construction, $[G]$ is a stochastic matrix: $\forall u \in V$, $\sum_{v \in V} g_{u,v} = 1$. The probability $P^t_G(u \rightsquigarrow v)$ of a walker starting on vertex u to reach a vertex v after t steps is:

$$P^t_G(u \rightsquigarrow v) = ([G]^t)_{u,v} \tag{1}$$

One can then prove [7], with the Perron-Frobenius theorem [17], that if G is connected (i.e., there is always at least one path between any two vertices), reflexive and

[1] i.e. each vertex is connected to itself. If such self-loops do not exist in the data, they may be generally added without loss of information.

undirected, then $\forall u, v \in V$:

$$\lim_{t\to\infty} P_G^t(u \rightsquigarrow v) = \lim_{t\to\infty} ([G]^t)_{u,v} = \frac{d_G(v)}{\sum_{x\in V} d_G(x)} \tag{2}$$

It means that when t tends to infinity, the probability of being on a vertex v at time t does not depend on the starting vertex but only on the degree of v. In the following we will refer to this limit as $\pi_G(v)$. If G is composed of several connected components then for any pair (u, v) of vertices, we have two possible cases:

- u and v are in the same connected component $G' = (V', E')$, with $V' \subseteq V$ and $E' \subseteq E$, then equation 2 applies to this subgraph:

$$\lim_{t\to\infty} P_G^t(u \rightsquigarrow v) = \lim_{t\to\infty} ([G]^t)_{u,v} = \frac{d_G(v)}{\sum_{x\in V'} d_G(x)} \tag{3}$$

- u and v are in distinct components, then for all t, $P_G^t(u \rightsquigarrow v) = 0$, therefore $\lim_{t\to\infty} P_G^t(u \rightsquigarrow v) = 0$.

So the probability $P_G^t(u \rightsquigarrow v)$ converges to a limit that only depends of vertex v degree. However the way this probability converges to the limit heavily depends on the topology of the graph between the two vertices. If u and v are connected by many short paths the probability will converge to the limit by above, whereas if there is no short path between the two vertices it will converge to the limit by below. Indeed when t is small the more interconnections there are between u and v, the higher the probability of reaching v from u. Therefore we define the t-confluence $\Gamma(G, u, v, t)$ between two vertices u, v on a graph G as follows:

$$\Gamma(G, u, v, t) = \begin{cases} \dfrac{P_G^t(u \rightsquigarrow v)}{\pi_G(v)} & \textbf{if } u \text{ and } v \text{ are in the same connected component,} \\ 0 & \textbf{else.} \end{cases} \tag{4}$$

We propose to consider as "close" each pair of non adjacent vertices (u, v) having a t-confluence greater than 1. In other words, we consider u and v as close if the probability of reaching v from u in a t step random walk is greater than the probability to be on v after an infinite walk. (u, v) is then labeled $0?$. Conversely non-adjacent vertices (u, v) having a t-confluence smaller than 1 are labeled $0!$.

In order to measure the closeness of an edge (u, v), the t-confluence is computed on the graph G where the considered edge has been removed. This removal is important, otherwise almost all edges would have a strong confluence, as the edge may be used by the random walker to go from u to v in few steps. The idea is to measure the closeness of the two vertices according to the graph structure and, this independently of the existence of an edge between them. Therefore, an edge (u, v) is labeled $1!$ if it has a t-confluence on the graph $G' = (V, E \setminus \{(u, v)\})$ greater than 1. In other words, without going through this edge, a random walker is more likely to be in v after t steps starting from u, than to be on v after un infinite walk. Conversely an edge (u, v) is labeled $1?$ if the t-confluence of (u, v) on the graph $G' = (V, E \setminus \{(u, v)\})$ is smaller than 1.

There are other possible ways of evaluating the closeness. Any measure of similarity between two vertices in a graph may be use, and in particular the ones developed to

address the problem of link prediction [9]. However we are interested in binary evaluation of similarity between to vertices, and there is rarely a natural threshold of gradual similarity measure. Also any robust graph clustering method [15] may be used: two vertices can be considered as close if and only if they are in a common cluster. However, note that the idea of short random walks proposed by [7] has been used in graph clustering method [13].

Illustration On the toy example of Figure 1, with the t-confluence labeling procedure, as one may expect, all edges are confirmed ($1!$), except edge d, and all non-edges are confirmed ($0!$), except the pair a. It has been verified for t between 2 and 20. With the triangle method, the results are the same except that many other non-edges are labeled $0?$. Indeed edge d creates many paths of length 2 between pairs of vertices that are not adjacent. Note that, for the same pairs of vertices, these paths of length 2 do not lead to a value of the t-confluence larger than 1.

Table 3 gives the number of pairs for each label on 3 different terrain networks[3,4] (the graph characteristics are given in table 10) and on one random network. Labels are computed according to the "triangle" method, abd with the 5, 10 and 20-confluence methods. As can be seen, the orders of magnitude of the four different labeled categories of pairs of edges are similar with the different methods. We note that most of the edges are confirmed in the two first terrain networks. This is not the case on the 3rd one where only about one third are confirmed. This is due to the fact that the synonyms network extracted from Wiktionary is very incomplete [14], which is not the case for the two previous networks that are based on linguistic resources that have been established for a long time. In the case of random network, the reported results are the average of the results obtained for 20 random networks of the same size, and we can notice that almost none of the edges are confirmed.

Note that if a vertex is connected to a large part of all the vertices, the triangle method would abusively consider as close all the pairs of neighbors of this vertex. This would not be generally the case with the random walk method.

Note also that these labeling methods may be restricted if one know, for instance, that the graph is fully correct. Indeed it will mean that every edge exists even if it is not confirmed by the topology. Therefore the labeling procedure could then be only applied to non-edges, and all edges are labeled $1!$. Conversely, if one knows that the graph is complete, and thus all non-edges are certain and labeled $0!$, while edges are labeled according to the graph topology.

3 Comparing graphs having the same vertex set

Comparing graphs is important in order to determine to what extent they contain the same information. In the following, we assume that the two graphs have *the same set of vertices*. In practice, this assumption mean that we compare two pieces of network information pertaining to the same set of objects or agents. For example, if a first graph represents friendship relation among a set of people, and a second one represents co-working relation inside the same set of people, one may be interested to know to what

extent these two relations are similar, or if one relation is included (or "almost" included) in the other.

In the following subsection, we propose a naive method for comparing two graphs by counting the number of matches "at the edge level". We shall see the limitation of this method. We then use the labeling method described in the previous section to compare graphs in a more robust way.

3.1 Classical agreement measure between edges

A simple method for comparing two graphs (having the same set of vertices) is to count on how many edges and no-edges they agree. Table 4 summarizes the 4 different cases: ok^+ is the number of edges present in both graphs, ok^- the number of non-edges present in both graphs, whereas $ko_1$is the number of pairs that are linked by an edge in the first graph, but not in the second one, and $ko_2$the number of pairs that are linked by an edge in the second graph, but not in the first one.

Table 4. Fusion of two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$

	E_1	$\overline{E_1}$
E_2	ok^+	ko_1
$\overline{E_2}$	ko_2	ok^-

We use Cohen's kappa coefficient [4] as a simple measure of agreement between two graphs. It is a inter-judge agreement measure. Here we consider each graph as a judge that annotates each pair of vertices either as "edge" or "non-edge". It is defined as follows:

$$Kappa(G_1, G_2) = \frac{p_0 - p_e}{1 - p_e} \tag{5}$$

with:

$$p_0 = \frac{1}{\omega}.(ok^+ + ok^-) = \frac{1}{\omega}.(|E_1 \cap E_2| + |\overline{E_1} \cap \overline{E_2}|) \tag{6}$$

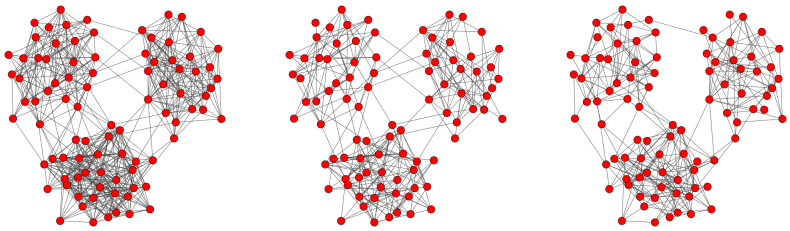
$$p_e = \frac{1}{\omega^2}.(|E_1|.|E_2| + |\overline{E_1}|.|\overline{E_2}|) \tag{7}$$

It has the advantage to take into account the agreement on edges (ok^+) and on non-edges (ok^-), without being influenced by the strong difference that exists in a terrain network between the size of these two sets (graphs are usually sparse, and thus there are many more non-edges than edges). Another alternative could be to measure the agreement only on edges, by using Jaccard coefficient

$$Jaccard(G_1, G2) = \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|} = \frac{ok^+}{ko_1 + ko_2 + ok^+} \tag{8}$$

between the two sets of edges. However we observe that these two measures behave in similar ways in the experimentations.

The column "edges" in Table 7 gives the values of the kappa and Jaccard coefficients on two pairs of synonymy networks. One can already note that this value are



(a) $g = (V, E)$ (b) $g_1 = (V, E_1)$ (c) $g_2 = (V, E \setminus E_1)$

Fig. 2. Artificial graphs with 3 clusters. g_1 and g_2 are subgraph of g and share the same 3 clusters but they have no edge in common.

low, which seems to attest a low agreement between synonymy networks. We comment these results more in detail in the section 3.3.

To demonstrate that the two above coefficients alone are insufficient for accounting for a global topological similarity of the graphs beyond the exact comparison pair of vertices by pair of vertices, we consider the following experiment. We build a graph $g = (V, E)$ with 3 groups of 30 vertices, where edges are built randomly with a probability 0.4 between two vertices of the same group, and 0.01 between vertices of two different groups. We then build a new graph $g_1 = (V, E_1)$ by randomly choosing half of the edges of g, and a new graph $g_2 = (V, E_2)$ such that $E_2 = E \setminus E_1$. These 3 graphs are plotted in Figure 2. The kappa measure between the two graphs g_1 and g_2 is negative (≈ -7.5 on 20 realizations) and the Jaccard measure equals 0. This would mean that these two graphs are completely dissimilar, which is true in the sense that they have no edges in common, however it is clearly wrong with respect to the topological "organization" they share. Indeed two vertices that are in the same group in the first graph will also be in the same group in the two other graphs. The above comparison methods have the drawback of only comparing graphs as "bag of edges", thus ignoring the topological structure created by these edges. We propose in the next section to use the labeling method presented in section 2.2 in order to propose a similar comparison method which does not suffer of this drawback.

3.2 Using the graph topology information

The labeling procedure described in Section 2.2 brings back topology information on each pair of vertices. We use this labeling procedure for comparing the two graphs,"pairs of vertices by pairs of vertices", without now missing the graph topology information. More precisely, if a pair is an edge confirmed by the structure in a graph (label 1!), but is a non-edge not confirmed by the structure in the other graph (label 0?) we consider that the two graphs do not disagree on this pair. Indeed the two vertices are topologically "close" in both graphs, even if they are adjacent in one, but not in the other. Similarly, if a pair is an unconfirmed edge in one graph (label 1?) and a confirmed non-edge in the other (label 0!), we consider that the two graphs agree on this pair as the two vertices are not "close" in any of the two graphs. The table 5 summarized the 16 different possible cases for a pair of vertices. We can now use the same kappa or Jaccard coefficients as in

the previous section, but now by counting as agreeing pairs those labeled $0?$ in a graph and $1!$ in the other, or $0!$ in one and $1?$ in the other.

Table 5. Comparison of two labeled graph

	$1!$	$1?$	$0?$	$0!$
$1!$	ok^+	ok^+	ok^+	ko_1
$1?$	ok^+	ok^+	ko_1	ok^-
$0?$	ok^+	ko_2	ok^-	ok^-
$0!$	ko_2	ok^-	ok^-	ok^-

$\Rightarrow$

	1	0
1	ok^+	ko_1
0	ko_2	ok^-

When we compare the two random graphs described in the previous subsection (see Figure 2) with this method that takes into account topology information, they appear to have a kappa (and a Jaccard) coefficient much higher than initially. Table 6 gives average comparison results for 20 random graphs using either the triangle or the 5-confluence method. We can see that they have no edges in common (by construction) but many of the edges present and confirmed ($1!$) in one graph are pairs of "close" vertices in the other ($0?$). We can also note that, as expected, the confluence method gives better results.

Table 6. Robust comparison of the graphs g_1 and g_2 of the Figure 2. Average value on 20 realizations.

(a) triangle, $kappa = 0.651$

	$1!$	$1?$	$0?$	$0!$
$1!$	0.0	0.0	127.3	58.9
$1?$	0.0	0.0	58.1	35.2
$0?$	128.7	59.8	390.2	302.3
$0!$	55.2	35.5	313.5	2440.2

(b) 5-confluence, $kappa = 0.881$

	$1!$	$1?$	$0?$	$0!$
$1!$	0.0	0.0	229.2	5.0
$1?$	0.0	0.0	31.9	13.3
$0?$	207.2	54.2	775.1	67.5
$0!$	3.1	14.4	58.1	2545.7

3.3 Comparison of synonymy networks

We illustrate the method proposed here on the comparison of pairs of synonymy networks. In such networks, one may expect that almost all edges are correct, even if few ones are "questionable", and that a large part of the non-edges are not related at all, even if some pairs of words are very close (but not really synonymous). We consider the networks $V.rob$ and $V.lar$, two synonymy networks between French verbs[2] and the networks $V.wikt$ and $V.pwn$, two synonymy networks between English verbs[3]. Table 7

[2] $V.rob$ and $V.lar$ are two synonymy networks between French verbs. There where digitalized from paper dictionaries (Robert and Larousse dictionaries) by an IBM/ATILF research unit partnership `http://www.atilf.fr/spip.php?article208`

[3] $V.wikt$ and $V.pwn$ two synonymy networks between English verbs. $V.wikt$ has been extracted from the English wiktionary by [14] whereas $V.pwn$ is built from Princeton Wordnet [5] synsets. A synset is a set of interchangeable words that denotes a meaning or a particular usage. The vertices of the network $V.pwn$ are the lemmas of the verbs present in Wordnet, and there is an edge $(x, y) \in E$ if and only if x and y belong to at least one common synset

gives the comparison results for the 2 French synonymy networks ($V.rob$, $V.lar$) and the 2 English synonymy networks ($V.wikt$ and $V.pwn$). Since these different networks do not have exactly the same lexical coverage, the comparison is based on the common sets of vertices. As can be seen, there is only a weak agreement between pairs of graphs, when they are compared by the classical agreement measure. This may not be expected, especially for the French graphs since they are obtained from authoritative general purpose dictionaries. Once the topology information is taken into account, we observe a strong agreement between the French graphs (up to 95 %) in the sense that almost all the edges of one graph are retrieved in the other as $0?$ labeled non-edges. In other words, most of the initial disagreements pertain to pairs of vertices that are close in both graphs (even if they are not adjacent in one). For the English graph, the agreement remains relatively weak. This is due to the fact that the wiktionary is very sparse, and not built at all in the same way as Wordnet. In Wordnet each edge reflects a common belonging to a synset, while the wiktionary graph edges are built by non expert contributors (without special care about synsets).

Table 8 provides similar comparisons on fictitious random graphs having the same overlaps as the French and English pairs of graphs previously considered. The obtained results strongly contrast with the previous ones, as expected. The random graphs still finally reach another form of strong agreement (now on "non-edges"), *but* only because the initial disagreement pertain to pairs that are *not* close in both graphs even if they are adjacent in one graph.

Table 7. Synonymy network comparison. Column "edges" gives the measures without the labeling procedure. $(0?, 1!)$ (resp. $(0!, 1?)$) indicates the number of pairs of vertices labeled $0?$ (resp. $0!$) in one graph and $1!$ (resp. $1?$) in the other.

		edges	triangle	5-confl.	10-confl.	20-confl.
$V.rob$ vs. $V.lar$	$Kappa$	0.518	0.876	0.937	0.953	0.946
	$Jaccard$	0.350	0.781	0.882	0.910	0.898
	$(0?, 1!)$	-	11769	16310	17401	16878
	$(0!, 1?)$	-	2860	1129	881	1050
$V.wikt$ vs. $V.pwn$	$Kappa$	0.202	0.498	0.600	0.636	0.673
	$Jaccard$	0.113	0.332	0.429	0.467	0.507
	$(0?, 1!)$	-	2511	3878	4485	5027
	$(0!, 1?)$	-	2246	1919	1667	1641

4 Fusing graphs

The same idea can also be used when merging two graphs. Let $G_1 = (V, E_1)$ and $G_1 = (V, E_2)$ be two graphs. A first type of merging could be to add, to the intersection of the two edge sets, the pairs of vertices that are labeled $1!$ in one graph and $0?$ in the other, i.e.,

$$E'_{\cap} = (E_1 \cap E_2) \cup \{\text{pairs labeled } (1!, 0?)\}$$

Table 8. Graphs comparison results on Erdös Rényi random network having the same initial overlaps as the real networks. Average on 20 realizations.

		edges	triangle	5-confl.	10-confl.	20-confl.
rob^R vs. lar^R	$kappa$	0.518	0.850	0.803	0.763	0.719
	$Jaccard$	0.351	0.739	0.671	0.617	0.562
	$(0?, 1!)$	-	801.4	1193.0	1097.8	0.2
	$(0!, 1?)$	-	16001.4	13734.2	12235.0	12209.9
$wikt^R$ vs. pwn^R	$Kappa$	0.203	0.716	0.592	0.541	0.595
	$Jaccard$	0.113	0.558	0.420	0.371	0.424
	$(0?, 1!)$	-	13.1	68.5	165.1	106.5
	$(0!, 1?)$	-	11865.0	10734.4	9919.4	10678.6

Table 9. Example of fusion of two social networks built from the e-mails of one of the paper's author for two different years. Pedigrees of these graphs are in Table 10: $mail10$ and $mail11$.

	1!	0?	0!
1!	65	16	32
0?	13	80	99
0!	38	110	1377

$|E_1 \cap E_2| = 65$
$|E'_\cap| = 94$
$|E'_\cup| = |E_1 \cup E_2| = 164$

Another type of merging (more tolerant) consists in removing the pairs of vertices labeled $1?$ in one graph and $0!$ in the other from the union of the edge sets of the two graphs, i.e.,

$$E'_\cup = (E_1 \cup E_2) \setminus \{\text{pairs labeled } (1?, 0!)\}$$

These two fusion procedures are such that the resulting edge sets $E'_\cap$ and $E'_\cup$ satisfy the following inclusions:

$$E_1 \cap E_2 \subset E'_\cap \subset E'_\cup \subset E_1 \cup E_2$$

This is illustrated with two graphs about e-mail relations between people. More precisely, we build an ego-centric social network from someone mailbox: each e-mail address u (which means more or less a person) is connected to another e-mail address v, iff u is the author of -at least- one mail having v as recipient ("To" or "CC"). It may be worth of interest to fuse such a graph built from all e-mails during a given year with the same graph built from e-mails of the previous year: we can then see which parts of the graph have been stable during these two years. The results corresponding to two social networks built from the e-mails of one person are shown in Table 9. Note that here the second fusing method $E'_\cup$, will give the same result as $E_1 \cup E_2$ since there is no edges labeled $1?$ as all the edges may be considered as "sure" since they rely on at least one existing e-mail. As can be seen 29 edges $(16 + 13)$ are restored on top of the edge sets intersection. Thus, they should not be count among the real change that took place between the two years.

Table 10. Pedigrees of 6 different terrain networks, n and m are respectively the number of vertices and edges, $\langle k \rangle$ is the average degree of vertices, n_{lcc} and m_{lcc} are the number of vertices and edges in the largest connected component,C is the transitivity coefficient of the graph, L_{lcc} is the average shortest path between any two nodes of the largest connected component, λ is the coefficient of the best fitting power law of the degree distribution and r^2 is the correlation coefficient of the fit.

	n	m	$\langle k \rangle$	n_{lcc}	m_{lcc}	C	L_{lcc}	λ	r^2
$V.rob$	7357	26567	7.48	7056	26401	0.12	4.59	-2.01	0.93
$V.lar$	5377	22042	8.44	5193	21926	0.17	4.61	-1.94	0.88
$V.wikt$	7339	8353	2.84	4285	6093	0.11	8.98	-2.40	0.94
$V.pWN$	11529	40919	8.16	9674	39459	0.24	4.66	-2.10	0.92
$mail10$	385	603	3.14	383	602	0.10	3.71	-1.11	0.73
$mail11$	391	671	3.45	389	671	0.06	3.32	-0.93	0.55

5 Related work

In the literature, the idea of graph comparison may refer to various problems and approaches. A first group of works deals with approaches that evaluate to what extent two graphs are isomorphic, or looks for approximate isomorphisms between two graphs. Measuring how two graphs are similar is a common problem for querying graph databases. Some methods [8, 18] use an edit distance between graphs. Other approaches measure the size of the maximal common subgraph[16, 20]. A related problem is to find a matching, or approximate matching between two graphs [11]. It consists in looking for a correspondence between vertices of one graph and vertices of the other such that the two graphs appears as similar as possible. The kappa and Jaccard measures (between not-labeled graphs) proposed in section 3.1 are comparable to such approaches in the "very" particular case where graphs have exactly the same vertices, and where each vertex cannot be put in correspondence with another one but itself. Besides, [2] proposes a different way of measuring graph similarity. This method gives a similarity score between any vertex of one graph and any vertex of a second graph. It applies between any pair of graphs, and does not consider any correspondence between vertices of the two graphs. So it may be applied when the two graphs are on the same set of vertices, however this knowledge is not taken into account by the method. A second group of works proposes to compare graphs by global statistical features [10], or compare graphs by measuring the number of occurrences of small particular sub-graphs [12]. To the best of our knowledge there was no work interested in comparing two graphs having the same set of vertices and taking into account the graph structure, if we except maybe [6].

6 Concluding remarks

This paper has presented a method that provides an augmented view of a undirected graph which acknowledges its underlying structure. This augmented view turns to be useful when comparing or fusing graphs, as illustrated in this paper, when we need to go beyond a purely "edge" by "edge" pairing. An obvious line for further research is the extension of the approach to weighted and/or directed graphs.

References

1. R. Albert and A. Barabási. Statistical mechanics of complex networks. 2001.
2. V. D. Blondel, A. Gajardo, M. Heymans, P. Senellart, and P. V. Dooren. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Rev.*, 46:647–666, April 2004.
3. B. Bollobas. *Modern Graph Theory*. Springer-Verlag, October 2002.
4. J. Cohen. A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.*, 20(1):37–46, 1960.
5. C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
6. B. Gaillard, B. Gaume, and E. Navarro. Invariants and variability of synonymy networks: Self mediated agreement by confluence. In *TextGraphs-6, ACL*, pages 15–23, 2011.
7. B. Gaume. Balades Aléatoires dans les Petits Mondes Lexicaux. *I3: Information Interaction Intelligence*, 4(2), 2004.
8. H. He and A.K. Singh. Closure-tree: An index structure for graph queries. In *Proc. of the 22th IEEE Int. Conf. on Data Engineering (ICDE)*, page 38, april 2006.
9. David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.
10. O. Macindoe and W. Richards. Graph comparison using fine structure analysis. In *Second IEEE Int. Conf. on Social Computing*, pages 193–200, aug. 2010.
11. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 117–128, 2002.
12. R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004.
13. P. Pons and M. Latapy. Computing communities in large networks using random walks (long version). *Journal of Graph Algorithms and Applications (JGAA)*, 10(2):191–218, 2006.
14. F. Sajous, E. Navarro, B. Gaume, L. Prévot, and Y. Chudy. Semi-automatic enrichment of crowdsourced synonymy networks: the wisigoth system applied to wiktionary. *Language Resources and Evaluation*, pages 1–34. (to appear).
15. S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.
16. H. Shang, K. Zhu, X. Lin, Y. Zhang, and R. Ichise. Similarity search on supergraph containment. In *Proc. of the 26th IEEE Int. Conf. on Data Engineering (ICDE)*, pages 637–648, march 2010.
17. G. W. Stewart. Perron-frobenius theory: a new proof of the basics. Technical report, College Park, MD, USA, 1994.
18. Y. Tian and J.M. Patel. Tale: A tool for approximate large graph matching. In *Proc. of the 24th IEEE Int. Conf. on Data Engineering (ICDE)*, pages 963–972, 2008.
19. D. Watts and S. Strogatz. Collective dynamics of'small-world' networks. *Nature*, 393:440–442, 1998.
20. X. Yan, P.S. Yu, and J. Han. Substructure similarity search in graph databases. In *Proc. of the 2005 ACM Int. Conf. on Management Of Data (SIGMOD)*, pages 766–777, 2005.